# Paper Implementation: *Fine-tune BERT for Extractive Summarization [1]*

**Jose Antonio Lorencio Abril, Sayyor Yusupov**

Machine Learning, M2 BDMA

Université Paris-Saclay, CentraleSupélec

Fall 2023

# Table of Contents

# Table of Contents

# Introduction

**Sample Text:**
Global warming is a significant challenge facing the world today. It's primarily caused by carbon emissions. These come from various sources, including transportation, industrial processes, and deforestation. The effects of global warming include rising sea levels, extreme weather events, and impacts on wildlife and ecosystems.

**Abstractive Summary:**
The world is facing the challenge of global warming due to carbon emissions leading to rising sea levels and severe weather changes.

**Extractive Summary:**
Global warming is a significant challenge facing the world today. It's primarily caused by carbon emissions.

# Introduction

**sum of sums**

The tokens are converted to an embedding space. There are two approaches for this task: extractive and abstractive summarization. Summarization is a popular NLP task, since the ability to summarize a text is inherently human and important for text understanding.

**take N sents**

The second approach, *take_N_sents*, involved extracting the best N summary sentences from each chunk and joining them together. The tokens are converted to an embedding space. A binary classifier module is added after BERT processing of the input, to assign the likelihood of belonging to the summary to each sentence, represented by its associated [CLS] token.

# Table of Contents

# BERT

- NLP model developed by Google in 2019 [2]
- **Main capability**: understand the context of a word in a sentence using information from both directions.
- The flow of an input text is as follows:
  1. Tokenize and add [CLS] and [SEP] tokens (maximum 512 tokens)
  2. Add segment embeddings (determined by [SEP] tokens) and positional embeddings
  3. Tansformer encoder processing
  4. Outputs 768-dimensional vectors, representing the contextualized word embeddings for each input token

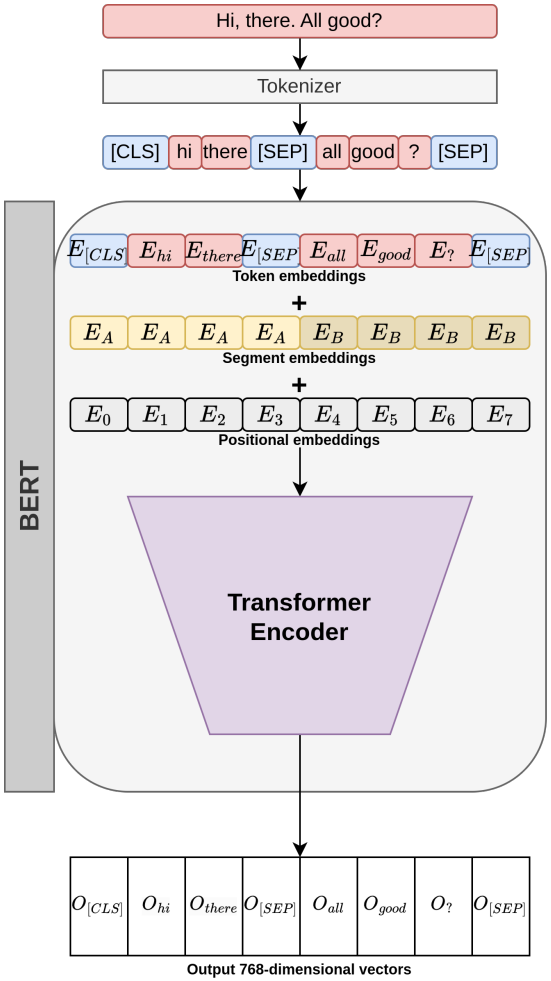



Figure: BERT Architecture

# BERT Pre-Training Tasks

**Masked Language Modeling (MLM)**
MLM randomly masks words in a sentence and the model predicts the masked word.

*Example:*
Original: The cat sat on the mat.
Masked: The cat sat on the [MASK].
Prediction: The cat sat on the mat.

**Next Sentence Prediction (NSP)**
NSP involves understanding the relationship between two sentences. BERT predicts if the second sentence logically follows the first.

*Example:*
Sentence A: I have a new laptop.
Sentence B: It works very fast.
Prediction: Is Next Sentence

Thanks to these two tasks, and specially leveraging NSP, BERT can be fine-tuned for extractive summarization.

# Table of Contents

# BERTSUM

- All sentences are separated with a [SEP] token $\rightarrow$ enforce segment encoding
- A [CLS] token is added at the beginning of each sentence $\rightarrow$ serves as *tabula rasa* to learn the new summarization capability
- A binary classifier is added at the end, to classify each [CLS] token as belonging to the summary or not

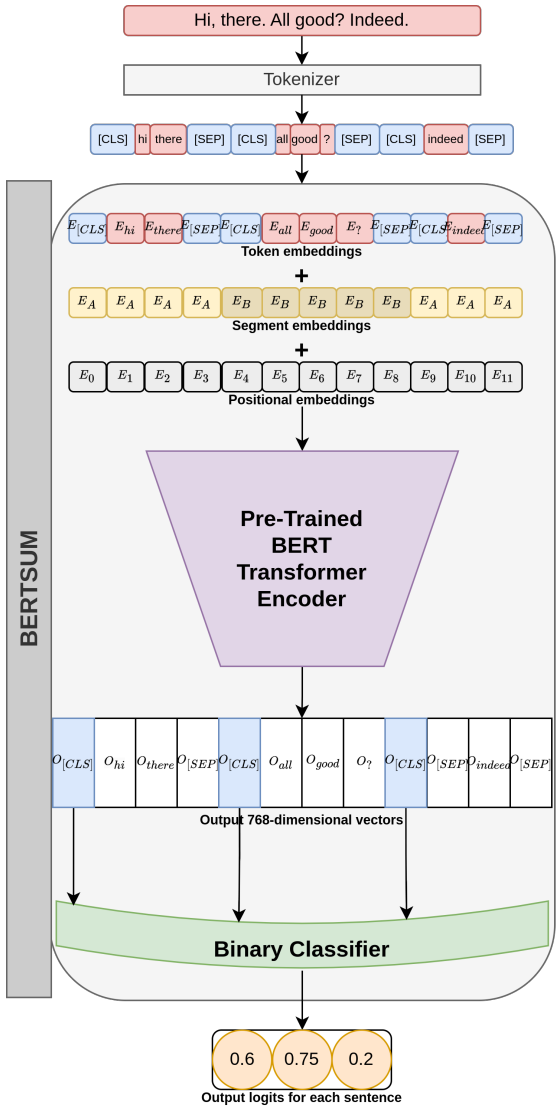



Figure: BERTSUM Architecture

# BERT vs BERTSUM

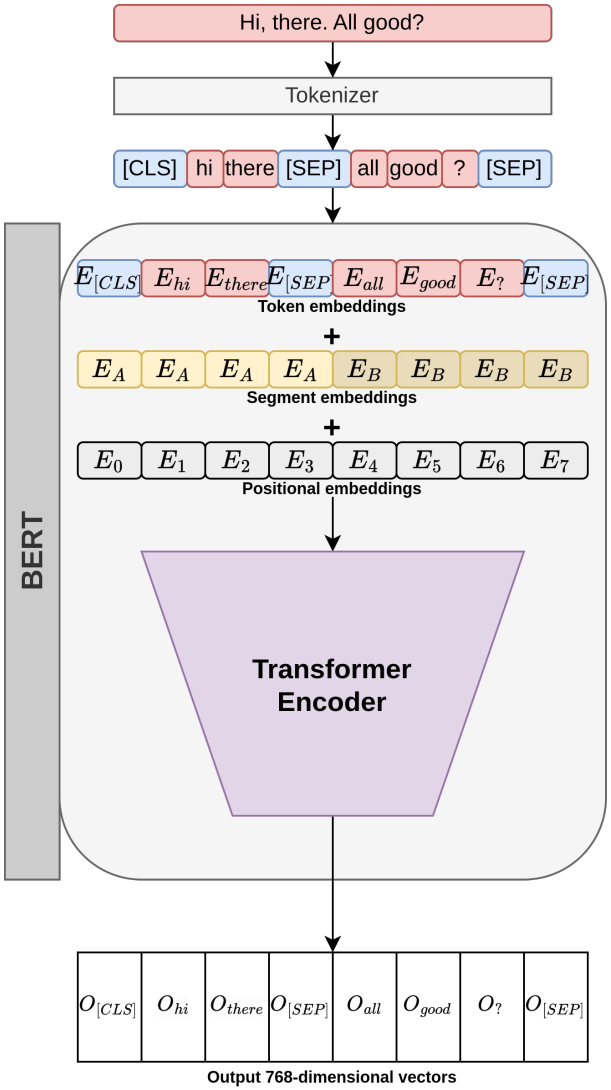



Figure: BERT Architecture

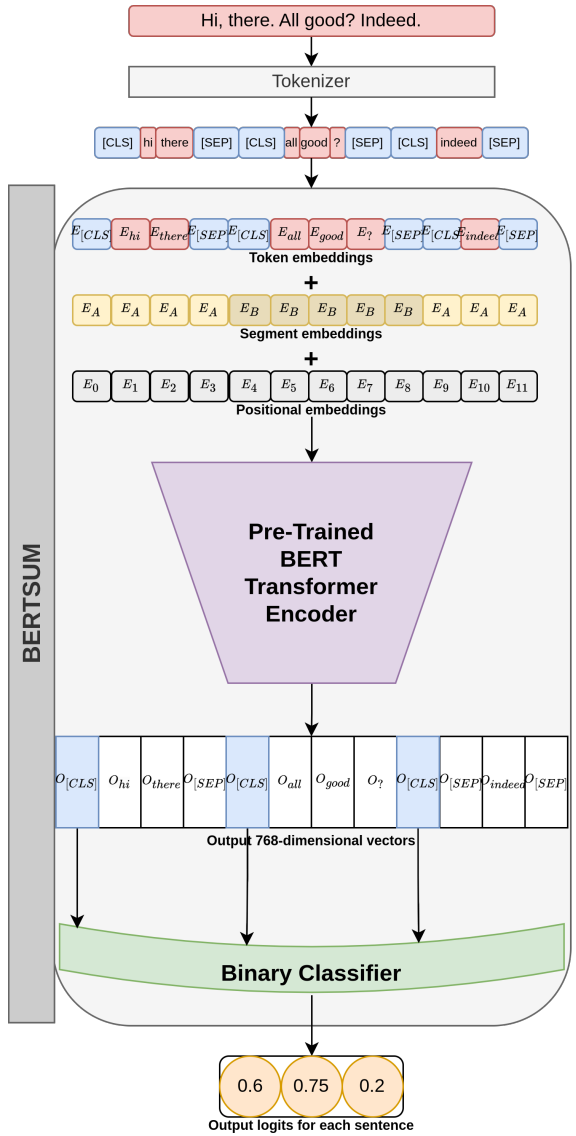
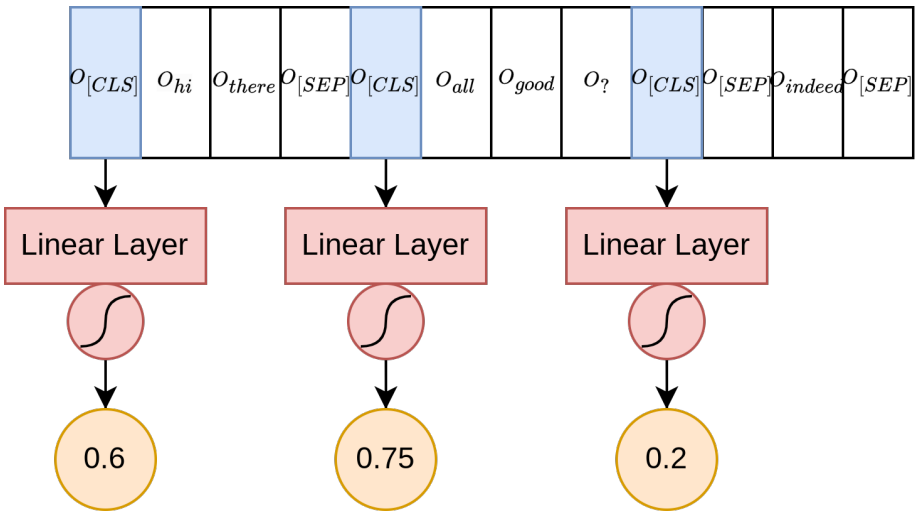
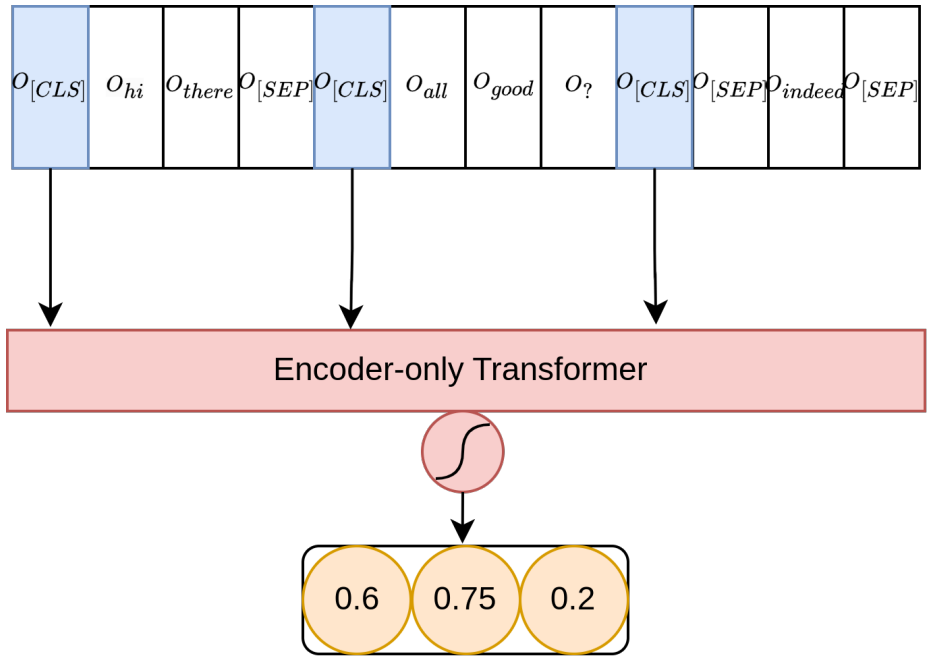



Figure: BERTSUM Architecture

# The Binary Classifier




(a) Linear classifier




(b) Encoder-only Transformer classifier

Figure: The different binary classifiers

# Table of Contents

# Datasets

- *Dialogsum* [3]: dialogs between two persons. Short (usually less than 512 tokens)

  *Example*: 'Jose: Hi, man. How are you. Sayyor: Hey! All good.'

- *arxiv-summarization* [4]: a set of research papers from arxiv, with the paper full text and its abstract. Long (several times the 512 tokens size)

  *Example*: Additive models @xcite provide an important family of models for semiparametric regression or classification. Some reasons...

# Table of Contents

# Evaluation Metrics

- **ROUGE-1**: measures the overlap of unigrams (single words) between the generated text, and the reference text.
- **ROUGE-2**: measures the overlap of bigrams (two-word sequences) between the generated text, and the reference text.
- **ROUGE-Lsum**: ROUGE-L finds the longest common subsequence between the generated and reference texts. ROUGE-Lsum is a variant of this, which aims at the sentence level. It's like applying ROUGE-L to each sentence individually, instead of to the whole text. The scores for all sentences are averaged at the end of the process.

# Table of Contents

# Pre-Processing

We need to obtain the extractive summaries. For this:

1. We use the given abstractive summary as a baseline.
2. Compute an extractive summary in a greedy fashion:
    1. We want to take $n$ sentences
    2. We do $n$ passes through the document sentences
    3. In each pass, we select the sentence that maximizes the increase in ROUGE score

# Long Texts

- Divide the document into chunks
- Each chunk is less than 512 tokens and sentences are not truncated in the process
- Chunks shorter than 200 tokens are discarded
- Extractive summaries are generated for each chunk as explained before

To process them with our model, we implement two approaches:

1. *sum_of_sums*:
    1. Process each chunk with BERTSUM
    2. Gather the summaries
    3. Process the gathered summaries with BERTSUM
2. *take_N_sents*:
    1. Process each chunk with BERTSUM
    2. Store all logits
    3. Take the top $N$ logits among all chunks as summary

# Table of Contents

# Training

- Optimizer: Adam with weight decay fix, with starting learning rate $lr = 10^{-5}$.
- Epochs: 10 and 100 epochs with *DialogSum*, 10 epochs with *Arxiv-Summarization*.
- Batch size: 32.
- Loss function: binary cross-entropy.
- Weights initialization:
  - BERT weights are the pre-trained weights.
  - Classifier weights are initialized randomly.

# Trained Models

We train six different models, belonging to two variants:

- BERTSUM-Dialog: trained on 12460 training instances, with 500 validation instances and 1500 test instances. Both using the linear and transformer classifier are trained for 10 and 100 epochs.
- BERTSUM-Arxiv: trained on around 167000 training instances, with 16700 validation instances and 16700 test instances. These instances, article chunks, were obtained from 10000 training, 1000 validation, and 1000 test articles. Both using the linear and transformer classifiers are trained for 10 epochs.

# Training outcomes

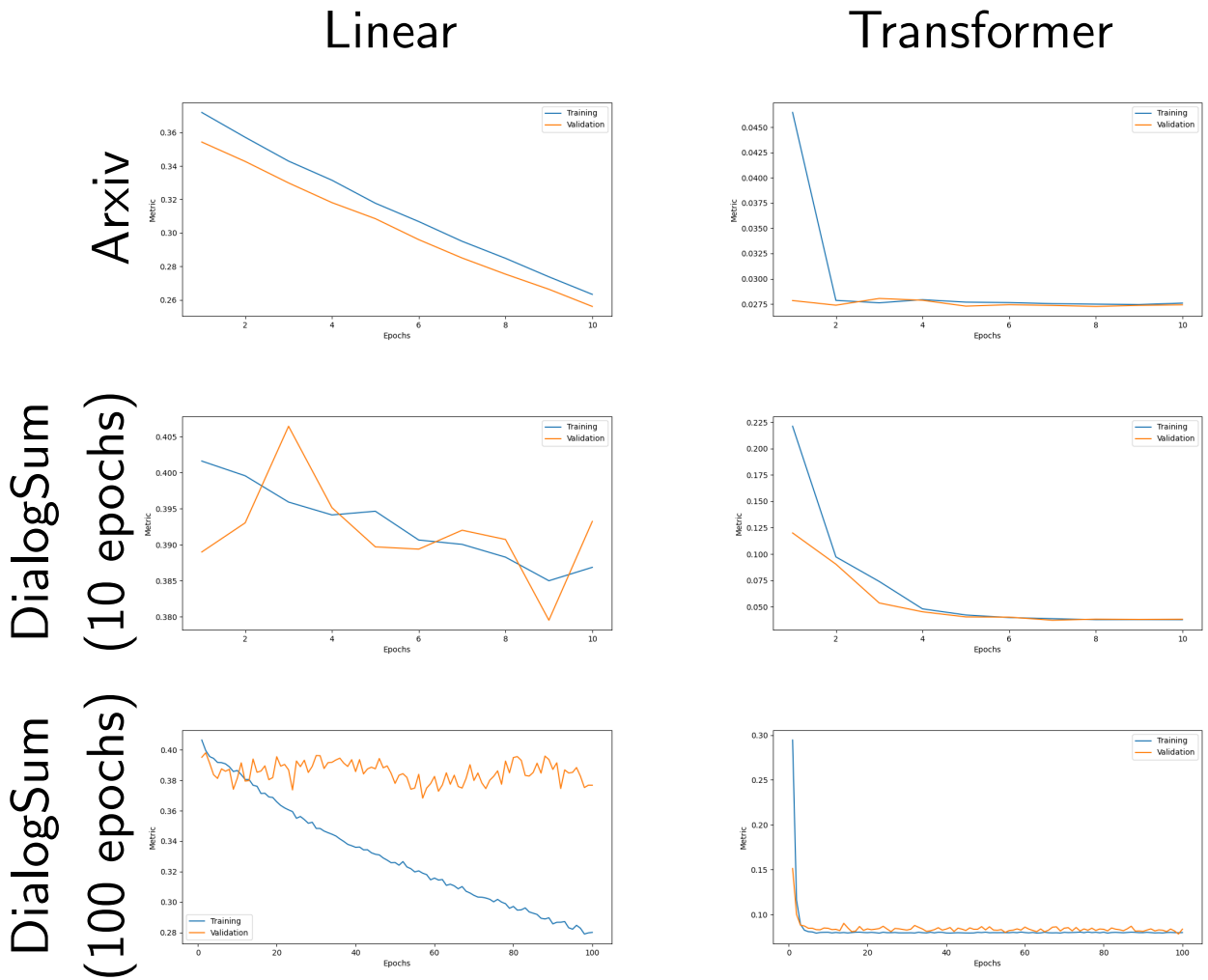



Figure: BCE-Loss evolution during training and validation

# Observations

- DialogSum data seems to provide most gains at the beginning
- Training with Arxiv data, however, seems to show steady improvement for both training and evaluation datasets when using the linear model. This is not as clear with the transformer model (**however**: higher cost for longer training)
- The BCE-Loss with the transformer model is around 10 times lower than with the linear model

# Table of Contents

# Experiments and Results

A. Run each model in its respective test dataset. Run it also in the other test dataset (arxiv papers are pre-processed, chunked)
B. With each of the models, we process the arxiv papers completely, without preprocessing. We process them using the two different strategies

We compare the predicted summaries and the base summaries obtained before with the ROUGE scores, averaging on the full datasets.

# Experiment A: Results

| Trained Dataset # epochs | Model Type | Testing Dataset | ROUGE-1 | ROUGE-2 | ROUGE-Lsum |
|---|---|---|---|---|---|
| Arxiv 10 epochs | Linear | Chunks | 0.434 | 0.293 | 0.346 |
| | | DialogeSum | 0.179 | 0.118 | 0.148 |
| | Transformer | Chunks | 0.437 | 0.301 | 0.349 |
| | | DialogueSum | 0.186 | 0.122 | 0.141 |
| DialogueSum 10 epochs | Linear | DialogueSum | 0.420 | 0.294 | 0.325 |
| | | Chunks | 0.439 | 0.298 | 0.348 |
| | Transformer | DialogueSum | 0.375 | 0.244 | 0.294 |
| | | Chunks | 0.438 | 0.295 | 0.349 |
| DialogueSum 100 epochs | Linear | DialogueSum | 0.409 | 0.282 | 0.318 |
| | | Chunks | 0.436 | 0.296 | 0.347 |
| | Transformer | DialogueSum | 0.139 | 0.068 | 0.114 |
| | | Chunks | 0.424 | 0.287 | 0.363 |

Table: Results of the 6 models on the 2 testing datasets.

# Experiment A: Analysis

- Models trained on the DialogueSum dataset generalize better to the unseen data
- Even, the models obtain better result in the unseen data: we believe that the dialogues are harder to summarize, and enforce the model to learn better the task
- The transformer model performs better in the Arxiv dataset
- The linear model performs better in the DialogueSum dataset
- The transformer model, when trained for 100 epochs on the DialogueSum dataset, obtains very poor results in this test set
- An interesting observation: the linear model trained on DialogueSum for 100 epochs obtains better results in the Arxiv data than the linear model trained on the Arxiv data

# Experiment A: A closer look

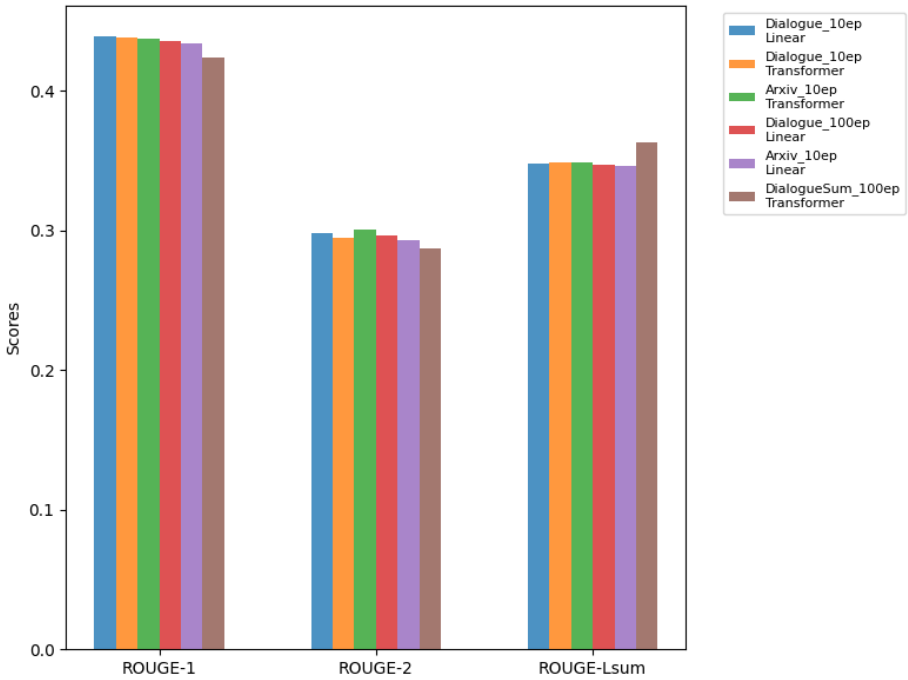



Figure: Arxiv Chunks Testing Dataset

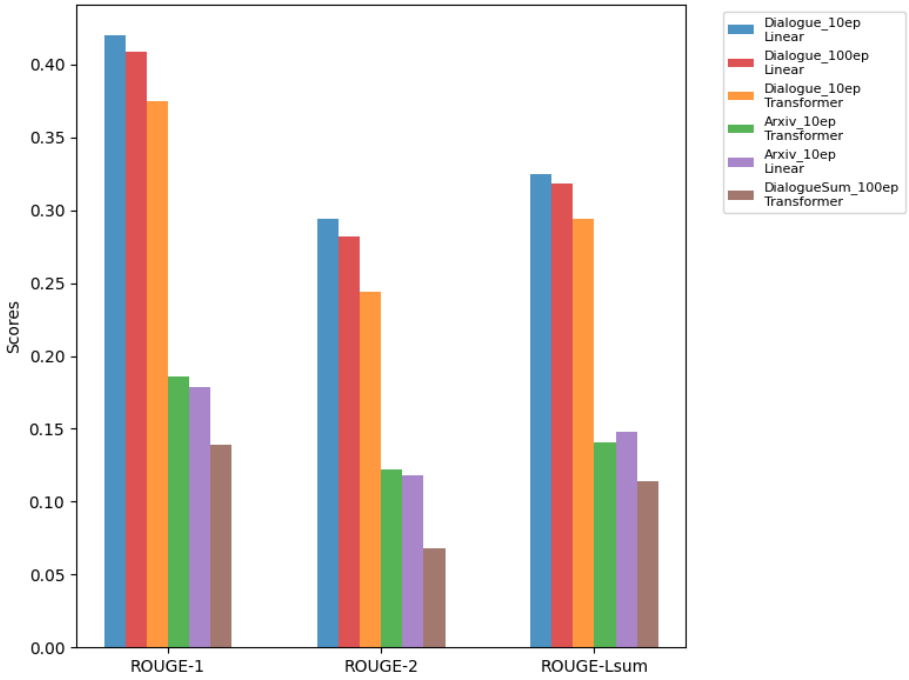



Figure: DialogueSum Testing Dataset

# Experiment B: Results

| Trained Dataset # epochs | Model Type | Predicting Strategy | ROUGE-1 | ROUGE-2 | ROUGE-Lsum |
|---|---|---|---|---|---|
| Arxiv 10 epochs | Linear | sum_of_sums | 0.350 | 0.118 | 0.234 |
| | | take_N_sents | 0.375 | 0.132 | 0.249 |
| | Transformer | sum_of_sums | 0.350 | 0.117 | 0.233 |
| | | take_N_sents | 0.352 | 0.117 | 0.239 |
| DialogueSum 10 epochs | Linear | sum_of_sums | 0.345 | 0.110 | 0.227 |
| | | take_N_sents | 0.318 | 0.092 | 0.211 |
| | Transformer | sum_of_sums | 0.340 | 0.108 | 0.227 |
| | | take_N_sents | 0.347 | 0.116 | 0.237 |
| DialogueSum 100 epochs | Linear | sum_of_sums | 0.350 | 0.117 | 0.234 |
| | | take_N_sents | 0.331 | 0.099 | 0.221 |
| | Transformer | sum_of_sums | 0.310 | 0.088 | 0.208 |
| | | take_N_sents | 0.338 | 0.106 | 0.230 |

Table: Performance of Prediction Strategies on Testing Dataset of the Whole Articles

# Experiment B: Analysis

- 20% or more decrease in performance from chunks dataset, expected as a harder task
- Usually take_N_sents is slightly better
- Arxiv-trained models perform best, but not much difference, others can generalize well
- Transformer models not trained on the data perform slightly worse than the rest

# Experiment B: A closer look

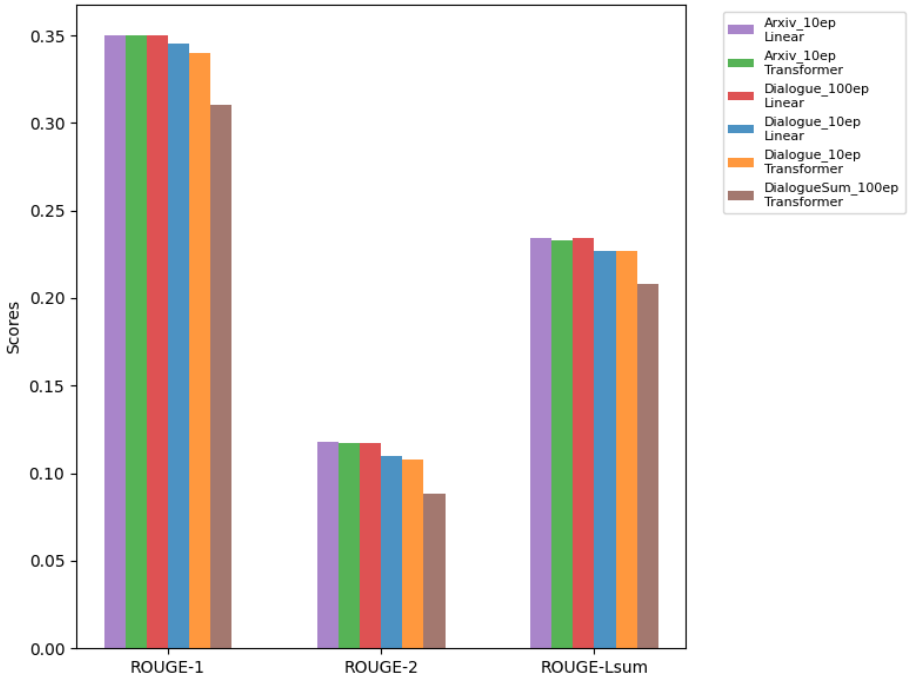



Figure: sum_of_sums on Full Arxiv

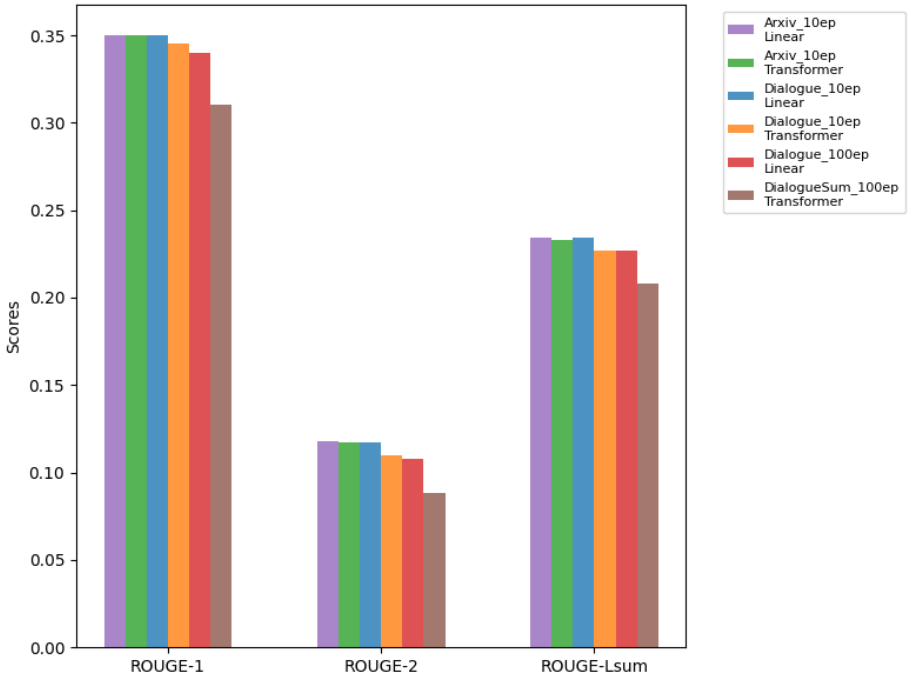



Figure: take_N_sents on Full Arxiv

# Table of Contents

# Bibliography I


[1] Yang Liu. *Fine-tune BERT for Extractive Summarization*. 2019. arXiv: 1903.10318.

[2] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805.

[3] Yulong Chen et al. "DialogSum: A Real-Life Scenario Dialogue Summarization Dataset". In: *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*. Online: Association for Computational Linguistics, Aug. 2021, pp. 5062–5074. DOI: 10.18653/v1/2021.findings-acl.449. URL: https://aclanthology.org/2021.findings-acl.449.

[4] Arman Cohan et al. "A Discourse-Aware Attention Model for Abstractive Summarization of Long Documents". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 615–621. DOI: 10.18653/v1/N18-2097. URL: https://aclanthology.org/N18-2097.